Blending, interpolating, synthesizing textures
Blend / interp: Which space is ‘linear’ ?

RGB or HLS or XYZ ? ( which color space ? which gamma ? )    I, E or magnitude ?
Lean: $\sigma$ or $\sigma^2$ ?  $\text{interp}(\sigma^2) \neq \text{interp}(\sigma)^2$  
Flakes ellipsoids: Q or $\Sigma = \frac{1}{Q}$ ?
Voxels: A, T, density ?

Never:  fields of (u,v), angles , phase (when wraps)
Issues: vectors

Raster or vector ? / Eulerian or Lagrangian ?
( BRDF: SH vs morphing...)

Raw data vs indirect
(high level handle):
histogram, probability...

[paper]
**Blending / splatting sprites or layers**

Sprites / splats ( / brushes )

Triplanar mapping

Contrast = $\sigma$. 
Blending / splatting sprites or layers

Contrast = $\sigma$.

$$\sigma^2(\alpha C_0 + \bar{\alpha} C_1) = E((\alpha C_0 + \alpha C_1)^2) - E^2(\alpha C_0 + \alpha C_1) = \alpha^2 \sigma_0^2 + \bar{\alpha}^2 \sigma_1^2 = (\alpha^2 + \bar{\alpha}^2) \sigma^2 \neq \sigma^2$$

$$\sigma^2(\Sigma \alpha_i C_i) = (\Sigma \alpha_i^2) \sigma^2$$

$\rightarrow \sigma(\frac{1}{N} \Sigma C_i) = \frac{\sigma}{\sqrt{N}}$  

NB: is law of large number : convergence to avg. (cf path tracing :-))
Blending / splatting sprites or layers

Contrast = \( \sigma \).

\[
\sigma^2(\alpha C_0 + \bar{\alpha} C_1) = E((\alpha C_0 + \alpha C_1)^2) - E^2(\alpha C_0 + \alpha C_1) = \alpha^2 \sigma_0^2 + \bar{\alpha}^2 \sigma_1^2 = (\alpha^2 + \bar{\alpha}^2) \sigma^2
\]

\[
\sigma^2(\sum \alpha_i C_i) = (\sum \alpha_i^2) \sigma^2
\]

\[
\rightarrow \sigma(\frac{1}{N} \sum C_i) = \frac{\sigma}{\sqrt{N}} \quad \text{NB: is law of large number : convergence to avg.} \quad \text{We want } \sigma! \]

Solution: make blending coefs such that  \( \sum \alpha_i^2 = 1 \)

\[
\rightarrow \text{simply normalized weights } \alpha_i \text{ by } \sqrt{\sum \alpha_i^2}! \quad (\text{Indeed, } \bar{C} + \frac{\text{Lerp}(C - \bar{C})}{\sqrt{\sum \alpha_i^2}}) \]

[ paper ] [ shadertoy ][2]
**Blending / splatting structured pattern**

Procedural, non-linear transform (clamp, LUT...): *naive blend → ghosting artefacts!*

\[ \sum \rightarrow \]

Non-linear: abs, shad

Solution between two images: morphing (disto mapping). won’t apply to procedural, + issues.
Blending / splatting structured pattern

Procedural, non-linear transform (clamp, LUT…) : naive blend → ghosting artefacts!

\[ \sum \rightarrow \]

Non-linear: abs, shad

Solution: Deferred non-linear part

\[ \text{noise } \{b_k\} \rightarrow \text{shad} \]

+ save some cost :-) 

NB: not only for procedural!

[paper] [shadertoy] [with advection]
**Space-Interpolating procedural param**

Want to modify the frequency of $\text{noise}(\text{freq}\cdot x)$ or $\sin(\text{freq}\cdot x)$ along space? or $\text{sound}(t)$

Bad idea: just replace $\text{freq}$ by $\text{freq}(x)$

**Expected:**

**Obtained:**
Space-Interpolating procedural param

Want to modify the frequency of $\text{noise}(\text{freq} \times x)$ or $\text{sin}(\text{freq} \times x)$ along space?
Bad idea: just replace $\text{freq}$ by $\text{freq}(x)$

Expected:

Obtained:

What you want is $\text{LUT}(\text{phase})$, with $\frac{\partial \text{phase}}{\partial x} = \text{freq}(x)$

\[ \rightarrow \text{phase} = \int_0^x \frac{\partial \text{phase}}{\partial x} \]

( if $\text{freq}$ is constant, is does give $\text{phase} = \text{freq} \times x$ )
Lookdev \( \downarrow \) mapping distortions

Texture advection, painterly animation… : keep the look despite distortions
Paradoxical requirements!
Lookdev ⊥ mapping distortions

Texture advection, painterly animation… : keep the look despite distortions
Paradoxical requirements!

Flow noise: time ⊥ space  [URL1, URL2] [ shadertoy ]
Texture advection

(a) Velocity field  (b) Input texture  (d) Naïve algorithm  (c) Our algorithm
Texture advection

(a) Velocity field
(b) Input texture
(d) Naïve algorithm
(c) Our algorithm

+ Procedural
+ Flownoise
Texture advection

Idea: regeneration if disto.

Eulerian way:

- 3-phased regenerated layer:
  
  [ shadertoy ]

  “motion without movement” illusion + contrast preservation
Texture advection

Idea: regeneration if disto.

**Eulerian way:** [papers: Eulerian]

- 3-phased regenerated layer:
  [shadertoy]

- Layers per duration (~ v-MIPmap) & masks

- Variant: time bidir in optical flow.
  [video Watercolor] [paper]
Texture advection

Idea: regeneration if disto

Lagrangian way:
Advect sprites

[ video QY ]
Other pattern conservations

- Motion without movement: [shadertoy]

- Seamless infinite/cyclic zoom: [shadertoy]

- Perceptions of order in noise: motion, 2, xor, symmetries, correlation...

- All-scale unit-integral noise: [shadertoy]
Details respect context
conserve something else

Distortion conserving the histogram: [shadertoy]
Details respect context
conserve something else

- Distortion conserving the histogram: [shadertoy]

- Influenced procedural: iterated Gabor noise renormalization
Synthesis:

1st, specification: what do we really want?

E.g. “I want to generate this”

stochastic - wavy - Fourier vs “features” vs specific - $\phi$

Fourier synthesis, Gabor, Perlin vs example-based vs RD, sym

None is good for all!

( free range vs) bounded vs target contrast?

How to normalize Fourier, Perlin? (but never clamp!)

Histogram? slopes? ‘profil’ of waves?

Sparse convolution vs Gabor

Props = globally, or in each sub-window (i.e. uniform)?

Spectrum prop implies (often) not what you think :-)
Fourier (including Gabor) always gives this: not this:

(contrast oscillations, Even in no LF)

Bad for LUT:

Challenges:
- Make criterions of different worlds talk together / add handles
- Controlling spectrum AND histogram/normalization
- Bridging between the look of different synthesis algorithms
- Understanding what is a texture :-)

→ my current research work around Gabor / Fourier / variance spectrum
early results...
Blending, interpolating, synthesizing textures